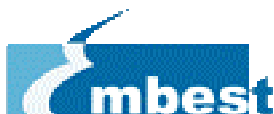
Embest S3CEV40 2004

用 户 手 册

Version 2. 1



深圳市英蓓特信息技术有限公司

地址: 深圳市罗湖区太宁路 85 号罗湖科技大厦 509 室（518020）

Tel: 86-755-25631365 86-755-25635656 Fax:86-755-25616057

E-mail: sales@embedinfo.com support@embedinfo.com

http://www.embedinfo.com http://www.embed.com.cn

目 录

前 言

本手册为英蓓特公司 Embest S3CEV40 开发板的用户手册，是 Embest S3CEV40 开发板的配套文档。该手册包含以下章节：

第一章　Embest S3CEV40 简介

第二章　安装使用

第三章　Embest S3CEV40 板硬件结构

第四章　Embest S3CEV40 软件系统

第五章　售后服务与技术支持

附录 A　跳线与开关设置

用户使用 Embest S3CEV40 开发板与该手册时，还可参考 Embest S3CEV40 开发板的电路原理图(发货光盘中提供)。

因时间仓促，手册中难免存在一些错误，敬请读者谅解，并欢迎指正，谢谢！

第一章 Embest S3CEV40 简介

1.1 概述

Embest S3CEV40 开发板是英蓓特公司推出的一款全功能 ARM 开发板，是 Samsung S3C44B0x 处理器设计参考、应用开发的最佳选择，是学习 ARM 和嵌入式教学的理想平台。

S3C44B0x 处理器基于 ARM7TDMI RISC CPU core，内置资源包括：

- 8KB cache/SRAM
- 扩展存储控制器（带 FP/EDO/SDRAM 控制器，片选逻辑）
- LCD 控制器（可直接控制 DSTN/STN 的各种灰度/256 彩色 LCD 屏，最大支持分辨率为 1600*1600）
- 2 通道 UART，波特率可高达 115.2kbps；内置 16Byte FIFO；兼容 IrDA1.0
- IIC 接口
- IIS 接口(音频数据接口)
- 8 通道 10bit ADC(采样速率为 100KSPS)
- 5 路 PWM 定时器&1 路内部定时器
- 71 个通用 I/O 端口和 8 个外部中断
- 2 路 GDMAs/2 路外围 DMAs
- 看门狗
- 电源控制器
- 实时时钟

Embest S3CEV40 开发板硬件系统包括存储器，I/O，数码管，液晶显示屏，触摸屏，键盘，音频输出，通讯接口包含串口、以太网接口，USB 接口，I2C 接口，高级扩展包括 IDE 硬盘，CF 卡，Flash 电子硬盘，是一款应用和接口非常全面的开发板。

Embest S3CEV40 开发板带有丰富的测试与驱动程序，并已移植好 uCOS-II 及 uCLinux 实时操作系统。Embest S3CEV40 开发板提供 20 针标准 JTAG 接口，用户可通过 JTAG 方式完全控制 CPU 和进行程序调试。

1.2 Embest S3CEV40 开发套件

完整的 Embest S3CEV40 开发套件出厂时配置为：

- Embest S3CEV40 开发板
- RS232 串口线
- 5V DC Power Adapter
- USB 线
- Embest S3CEV40 CD-ROM：

  - 开发板手册资料
  - 开发板电路原理图
  - 启动程序与各功能模块测试程序
  - 实时操作系统 uCOS-II 全部源程序

请在打开包装时检查以上货件是否齐全，若有遗漏，请联系您的销售商。

1.3 Embest S3CEV40 的特点

Embest S3CEV40 开发板硬件特性：

- 电源：DC5V1.5A 供电或由 USB 接 PC 供电；电源指示 LED 以及 500mA 保险丝
- 2 个串口，其中一个为简单接口，一个为全接线接口，可连接 RS232 MODEM
- 1M×16bit Flash
- 4×1M×16bit SDRAM
- 固态硬盘 16M×8bit
- 4Kbit IIC BUS 的串行 EEPROM
- LCD 及 TSP 触摸屏接口
- 外部 IDE 硬盘接口
- 4×4 用户键盘
- USB 设备
- 10M 以太网接口
- MICROPHONE 输入口
- IIS 音频信号输出口，可接双声道 SPEAKER
- 320*240 带触摸功能的显示屏(可选)
- 4 个 2×20PIN CPU 扩展接口
- 两个中断按钮，两个 LED
- 20 针 JTAG 接口
- 8 段数码管
- 复位开关

第二章 安装使用

本章主要是介绍 Embest S3CEV40 开发板如何安装，连接和进行基本的软硬件测试。

2.1 Embest S3CEV40 开发板结构

Embest S3CEV40 开发板共由三部分组成，主板，LCD 子板，键盘子板，开发板结构图如图 2-1 所示：

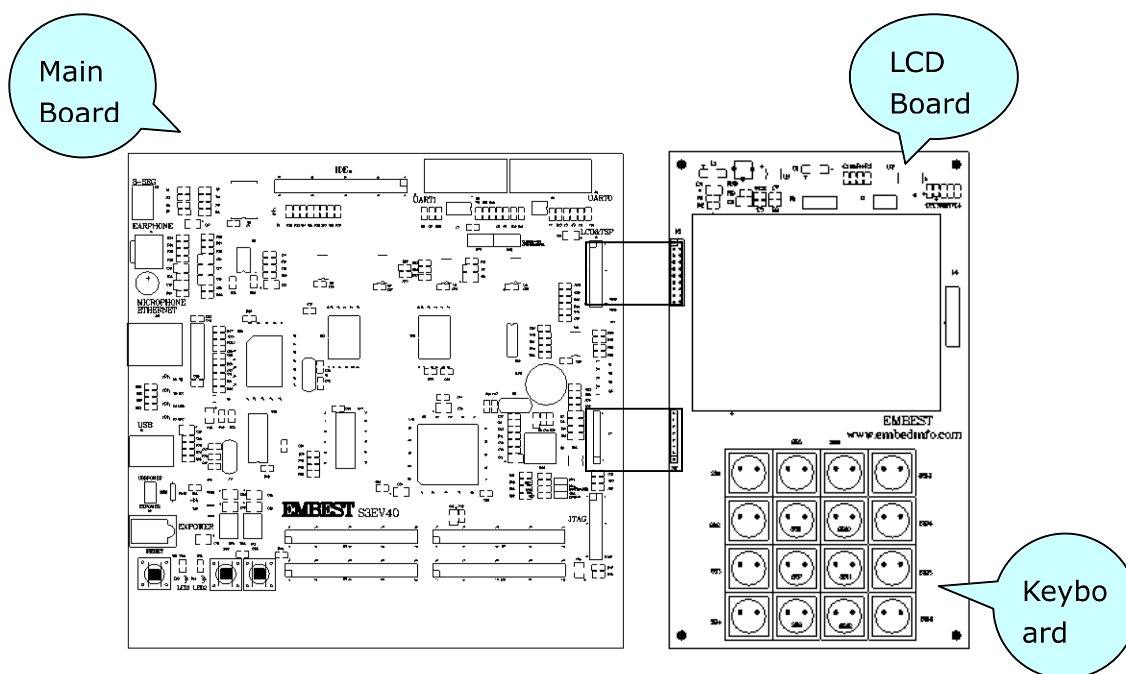


图 2-1 Embest S3CEV40 开发板整体示意图

其中，Embest S3CEV40 开发板主板的结构图如图 2-2 所示：

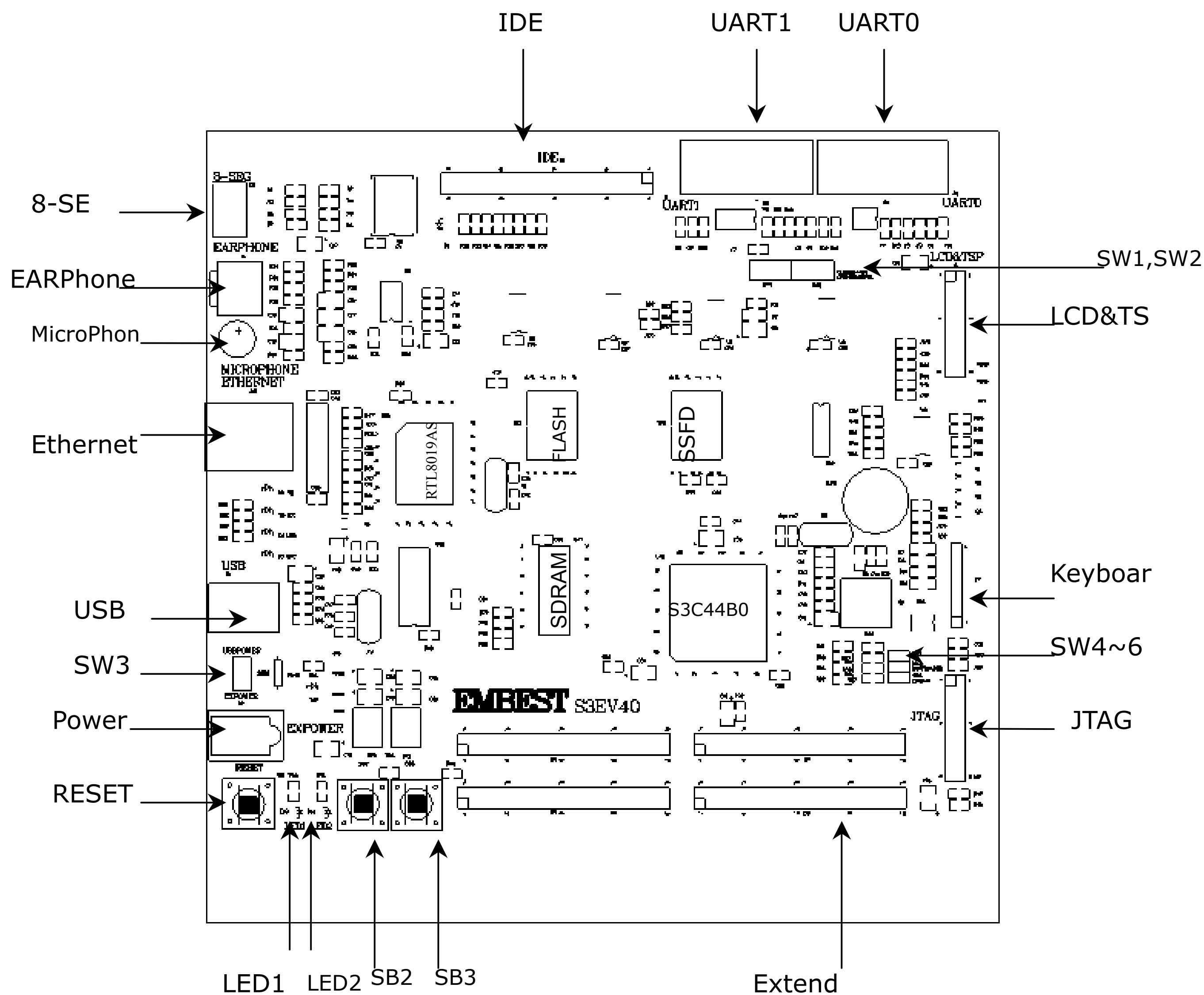


图 2-2 Embest S3CEV40 开发板主板结构图

2.2 Embest S3CEV40 开发板供电

Embest S3CEV40 开发板使用 5V DC 和 500mA 额定电流。开发板配套了专门变压器，用户可以使用变压器或者外接电源线给开发板供电。

2.3 Embest S3CEV40 开发板 JTAG 连接

图 2-3 是 Embest S3CEV40 开发板开发模型，主机通过 JTAG 仿真器连接开发板，主机运行开发环境，如 Embest IDE，可以直接通过仿真器下载，调试等。

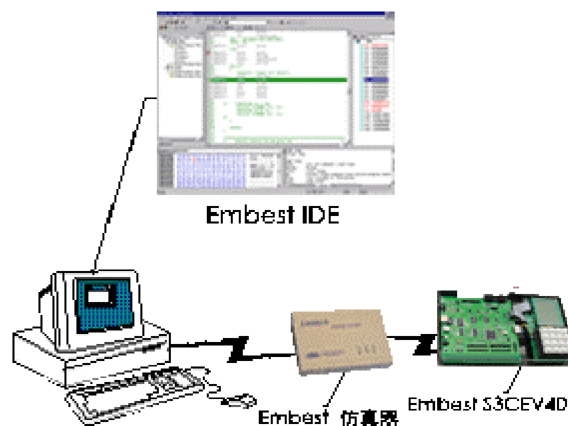


图 2-3 Embest S3CEV40 开发板开发模型

Embest S3CEV40 开发板提供了标准的 20 针 JTAG 接口，用户可以使用该板附带的 JTAG 线连接仿真器。

2.4 Embest S3CEV40 开发板串口连接

用户可以使用 Embest S3CEV40 开发板提供的串口线把开发板和 PC 主机连接起来，主机通过超级终端和开发板通讯，以方便进行前期的硬件测试。

串口线连接开发板的串口 UART0， 主机运行超级终端设置：串口 COM1 或者 COM2，波特率 115200，如下图 2-4 所示：



图 2-4 超级终端串口设置图

2.5 Embest S3CEV40 开发板硬件测试

英蓓特公司提供了该开发板的详细硬件测试程序，打开实验板附带的光盘，拷贝S3CEV40 例程文件夹到 EmbestIDE 安装目录\Samsung 文件夹下。运行相应的工作区文件，可以单独测试实验板的各部分硬件。

用户还可以使用英蓓特提供的出厂测试系统程序，按照以下步骤检测该板的硬件资源是否正常工作。

使用 RS232 标准串口线正确连接开发板（UART0）和主机 PC（COMx）；接通电源；开发板的数码管八段全亮；LED1、LED2 轮流闪烁（频率近 1Hz）；使用 PC 键盘操作；串口终端输出信息如图 2-5 所示：

```
Embest S3C44B0X Evaluation Board(S3CEV40)
*==========================================*
*=            Embest Info&Tech Co.,LTD.           =*
*==========================================*
*===========        R&D CENTER        ===========*
*========          86-755-25631365          ========*
*======          support@embedinfo.com          ======*
*===                        Version 2.1                        ===*
Please select test item:
1: 8LED test     2: LCD test       3: Keyboard test
4: Sound test    5: Timer test     6: Ethernet DHCP test
7: Flash test    8: IIC test       9: Ethernet TFTP test
0: TouchScreen test
>
```

图 2-5 开发板加电/复位后超级终端输出

用户可以通过 PC 键盘输入 0~9 来选择测试特定的硬件，其中：

- 1 8LED test ：8 段数码管测试。用户选择输入 1 ，超级终端输出如图 2-6 所示；开发板的数码管正常工作会自动从 0-F 计数。显示 0 – F 后返回初始状态。

```
> 1
Look at 8-segment Digit LED...
```

图 2-6 数码管测试超级终端图

- 2 LCD test ：LCD 液晶测试。用户选择输入 2 ，超级终端输出如图 2-7 所示；液晶显示屏输出矩形框和英蓓特公司名称如图 2-8 所示；同时返回初始状态。

```
> 2
Look at LCD...
```

图 2-7 LCD 测试超级终端图

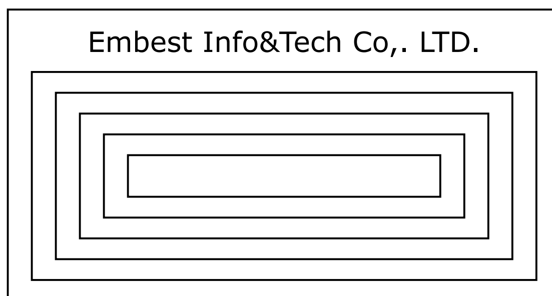


图 2-8 LCD 液晶画面图

- 3 Keyboard test ：键盘测试。用户选择输入 3 ，超级终端输出如图 2-9：

```
> 3
Please press one key on KeyBoad(4 x 4) and look at LED ...
```

图 2-9 键盘测试超级终端图

用户可以通过开发板的小键盘输入 0-F 的数值，开发板数码管会自动显示相应的数值。

- 4 Sound test ：音频测试。用户选择输入 4 ，超级终端输出如图 2-10：

```
> 4
  CODEC: Philips UDA1341 (U5)
  Please listen to sound.
  Press any key to exit...
```

图 2-10 音频测试超级终端图

音频正常输出时用户可以听到连续的“嘀嘀…”声；按下 PC 机任意键返回初始状态。

- 5 Timer test ：定时器测试。用户选择输入 5 ，超级终端输出如图 2-11 所示：

```
> 5
Timer Start, press any key to exit...
****
```

图 2-11 Timer 测试超级终端图

如果定时器工作正常，用户可以看到每隔一个固定时间就打印一个“*”；按下 PC 机任意键返回初始状态。

- 6 Ethernet DHCP ：动态配置 IP 地址。用户使用 Ethernet DHCP test 前，必须先在本地或网络 PC 机上运行 DHCP Server 程序或其他类似动态 IP 配置工具（由于本示例使用的 DHCP Server 软件是共享软件，所以附带光盘没有提供）；用户可以使用其它类似工具软件。按照软件正确配置 DHCP 服务后，设置参数完毕之后，用户在 PC 机超级终端输入选择 6 ；若没有动态 IP 配置软件在运行或运行不正常，超级终端输出如 2–12 所示，按下 PC 机 ESC 键返回初始状态；

```
> 6
 Waiting DHCP server to Respond.
 Press ESC key to exit ...
```

图 2-12 等待 DHCP Server

如果成功配置 IP 地址，即返回所配置的 IP，输出如图 2-13 所示.

```
Receive DHCP Message from server 192.192.192.88
Config local ip address 192.192.192.5          （所配置的 IP 地址）
```

图 2-13 DHCP 配置成功后输出

- 7 Flash test ：Flash 读写测试，用户输入选择 7 ，超级终端输出如图 2-14 所示：

```
> 7
  SST39VF160-90 (U12)
Write 0x000-0xff to flash address 0x30000...
Flash Write and Check Success!
```

图 2-14 Flash 测试超级终端图

往 Flash 中某地址空间顺序写入 0x00-0xff，并进行读出比较，成功后显示 Success 提示。按下 PC 机任意键返回初始状态。

- 8 IIC test：IIC 测试。用户输入选择 8 ，超级终端输出如图 2-15 所示：

```
> 8
IIC Test using AT24C04 (U18)...
Write char 0-f into AT24C04
Read 16 bytes from AT24C04
 0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
```

图 2-15 IIC 测试超级终端图

测试程序往 AT24C04 某址写入 0 - F，并从同一地址读出数值输出到超级终端。

- 9 Ethernet TFTP test ：TFTP 传输测试。输入选择 9 ，超级终端输出如图 2-18 所示。用户可先在主机端运行 TFTPDown 软件或其他 TFTP 下载软件配合开发板通讯。TFTPDown 软件可在开发板附带的光盘找到，名称为 TFTPDown.exe 。运行前需要用户在主机上配置静态 IP 地址，使用说明如下：

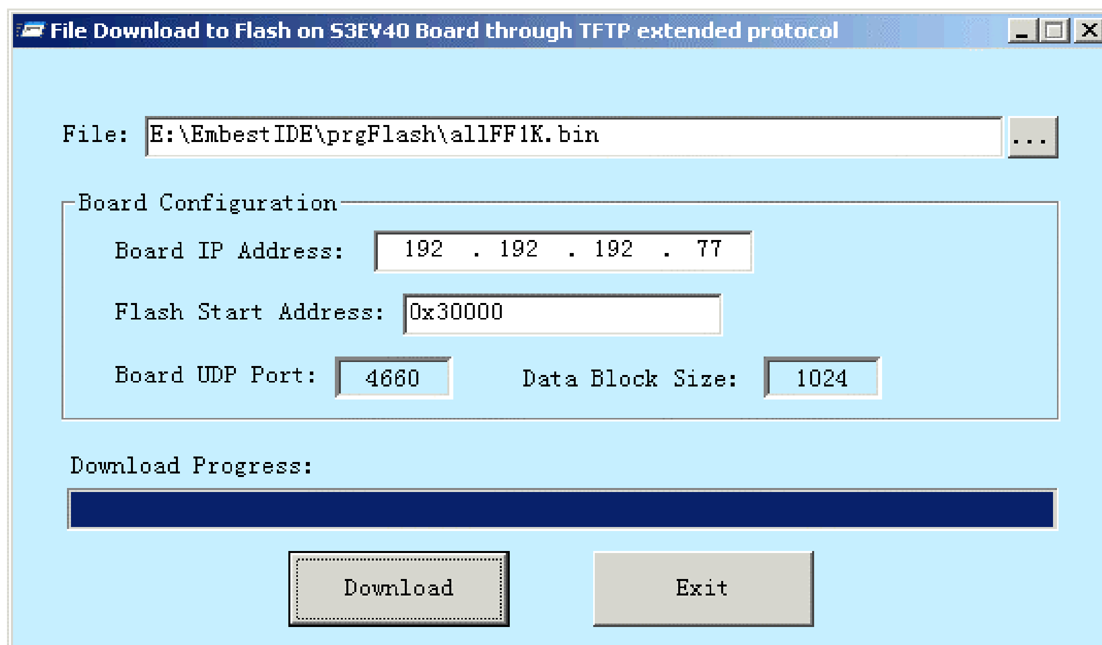
  1、MSDOS 状态下设置静态 IP 如下：

  arp -s 192.192.192.77 00-06-98-01-7e-8f　*设置静态 IP*

  arp –a *查看配置成功与否*

  2、运行 TFTPDown.exe 后按照实际情况进行配置

  主机运行 TFTPDown 的界面如图 2-16 所示；按下 PC 机 ESC 键返回初始状态。



allFF1K.bin 为测试文件，可以使用其他 bin 文件；用户可自定义下载 FLASH 地址。

图 2-16 TFTPDownload 运行界面图

用户按照上图参数设置完毕后，点击 Download 按钮，TFTP 正确运行会出现以下提示：

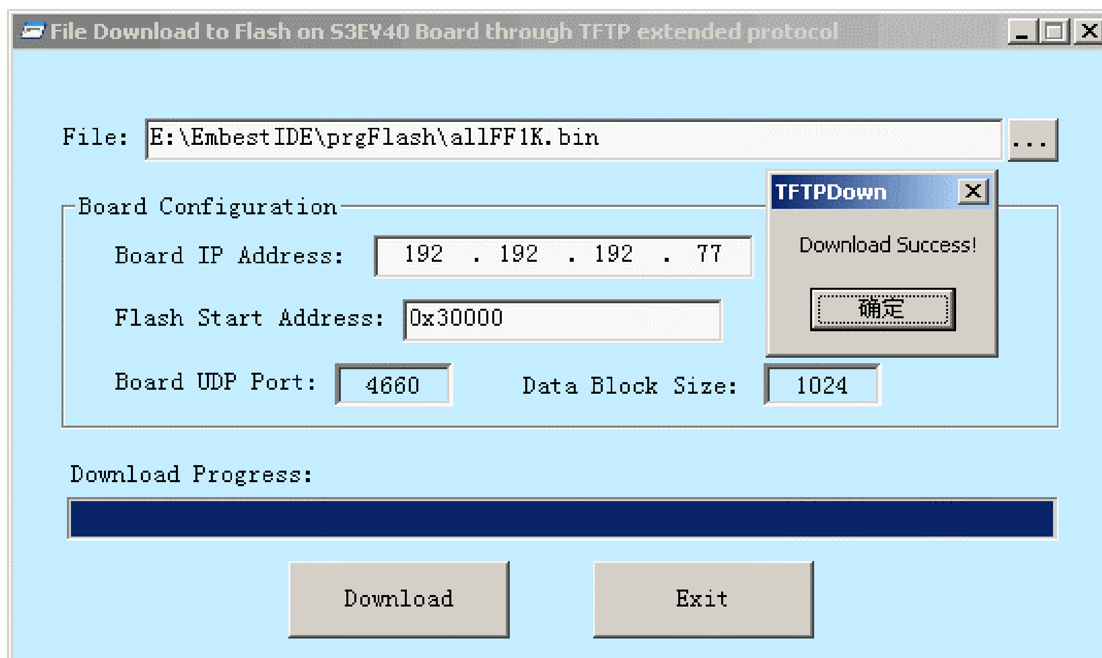


图 2-17 TFTP Download 测试成功图

当开发板所在的网络不能正常通信或开发板硬件不正常时，TFTP 文件下载会显示下载失败提示。用户可以先检查网络通信是否正常再进行调试，直到通信成功。

```
Do you want to configure local IP ?

 Y/y to configure local IP addr; D/d to use Default IP addr(192.168.0.200).

 Press any key to continue ...
( y ) Please input IP address(xxx.xxx.xxx.xxx) then press ENTER:   按下 Y 或 y

192.192.192.77   输入合法的 IP 地址（注意应与本机 PC 在同一网段内）
Manual Set local ip 192.192.192.77
Press any key to exit ...
```

图 2-18 Ethernet TFTP test 超级终端图

- 0 TouchScreen Test ：触摸屏测试。激活该项测试时，超级终端输出如图 2-19；触摸屏被分成十六格并显示 0~F；当用户触摸液晶屏时，通过串口在超级终端输出相应的坐标值（同时输出上一次触摸坐标；坐标原点靠近 4x4 键盘左上方）如图 2-20 所示。触摸屏理论上可以识别 1 个单位的量，建议用户使用触摸屏设计时以 10 个单位为识别间隔。出厂测试程序对触摸屏只完成数模转换部分测试，关于坐标的定位，请参照 TouchScreen_Test 例程。按下 PC 机任意键返回初始状态。

```
> 0
 Touch Screen coordinate Rang in:                  LCD 显示 16 小格
   (Xmin,Ymin) is :(xxx,xxx)                        触摸后串口输出
   (Xmax,Ymax) is :(xxx,xxx)                         所在坐标值
  To use current settings. Press N/n key.
  Want to Set Again(Y/N)? Y 或 y       按下后再触摸显示屏的任意对角进行坐标定位

Touch TSP's Cornor to ensure Xmax,Ymax,Xmax,Xmin
 User touch coordinate(X,Y) is :(0239,0679)          第一个顶角坐标
 User touch coordinate(X,Y) is :(0608,0303)          第二个顶角坐标
 Touch Screen coordinate Rang in:
   (Xmin,Ymin) is :(0239,0303)  (x1,y1)
   (Xmax,Ymax) is :(0608,0679) (x2,y2)
  To use current settings. Press N/n key.
  Want to Set Again(Y/N)? N 或 n        按下后用户可以触摸显示屏输出所在坐标值
* 如果定位范围误差过大，按下 Y 或 y 重新定位即可，正常 x2-x1 略大于 320；y2-y1
略大于 240
```

图 2-19 触摸屏坐标定位超级终端输出

```
Pixel: 320 X 240. Coordinate Rang in: (0,0) - (320,240)
LCD TouchScreen Test Example(please touch LCD screen)
press any key to exit...
X-Posion[AIN1] is 0097      Y-Posion[AIN0] is 0132
X-Posion[AIN1] is 0117      Y-Posion[AIN0] is 0132
```

图 2-20 触摸屏的坐标值输出

经过上面 10 个测试，开发板功能基本测试完毕，如果用户发现错误，请与英蓓特公司市场部联系以协助解决。

实验板加电/复位后，如果连接了液晶显示零，可以看到相应的信息：LCD 屏显示箭头图及测试功能选项；并通知用户使用 PC 键盘进行测试；配合 PC 串口操作，可以使用开发板用户键盘输入操作码，功能选项与串口操作一样。信息界面如下：

```
                    （箭头图形位置）
Embest S3C44B0X Evaluation Board(S3CEV40)
*====        Embest Info&Tech Co.,LTD.          ====*
Please select test item:
1: 8LED test     2: LCD test       3: Keyboard test
4: Sound test    5: Timer test     6: Ethernet DHCP test
7: Flash test    8: IIC test       9: Ethernet TFTP test
0: TouchScreen test
>
Note: Please connect UART0(115.2K,8,N,1) to PC COMx
```

图 2-21 显示屏正常加电/复位后信息输出

第三章 Embest EV44B0 板硬件结构

3.1 Embest EV44B0 开发板电路结构

Embest EV44B0 开发板电路结构如图 3-1 所示，图中包含了该板的所有主要功能模块。



图 3-1 Embest EV44B0 方框图

3.2 Embest EV44B0 核心电路

3.2.1 S3C44B0 处理器

Embest EV44B0 的核心是基于 ARM 公司 ARM7TDMI 内核的 16/32 位 RISC 处理器 -S3C44B0X。该处理器为手持设备和一般应用提供高性价比和高性能的微控制器解决方案。图 3-2 为 S3C44B0X 框图。为了降低整个系统的成本，S3C44B0X 还提供以下部件：8KB Cache、可选的内部 SRAM、LCD 控制器、2 通道 UART、4 通道 DMA、系统管理器（芯片选择逻辑、FP/EDO/SDRAM 控制器）、6 通道带 PWM 的定时器、I/O 口、RTC、8 通道 12 位 ADC、IIC/IIS 总线接口、同步 SIO 接口和成对时钟的 PLL。

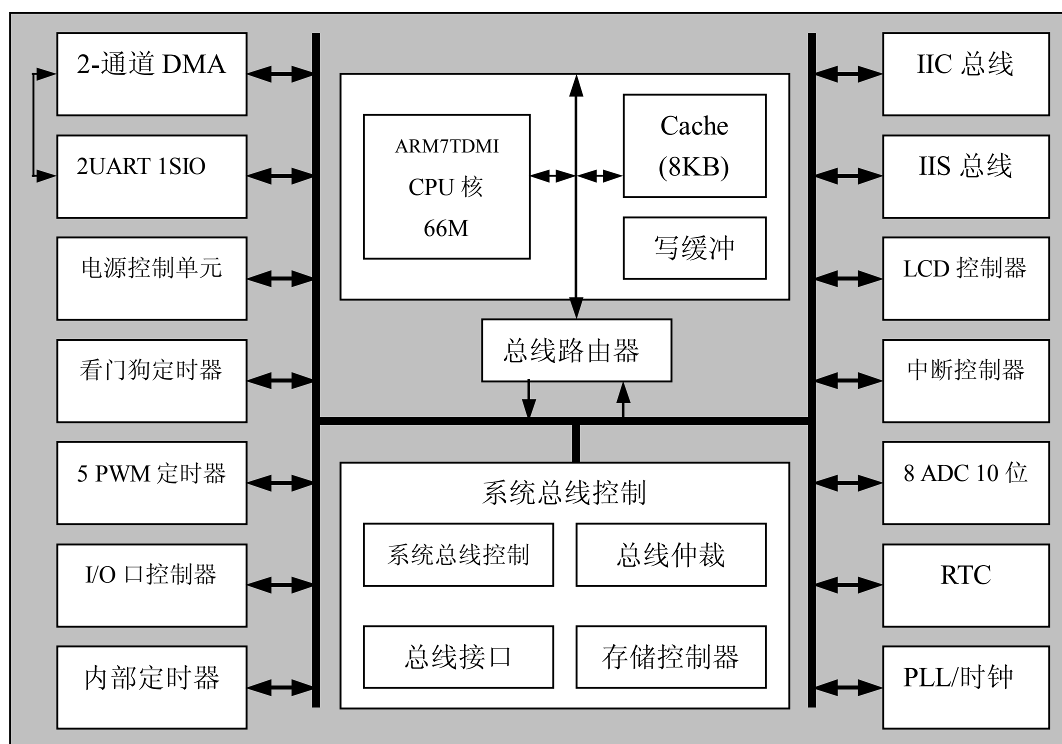


图 3-2 S3C44B0X 框图

3.2.2 存储系统

开发板上的存储系统包括一片 1M×16bit 的 Flash（SST39VF160）和一片 4M×16bit 的 SDRAM（HY57V65160B）。

如图 3-3 Flash 连接电路所示，处理器是通过片选 nGCS0 与片外 Flash 芯片连接。由于是 16bit 的 Flash，所以用 CPU 的地址线 A1-A20 来分别和 Flash 的地址线 A0-A19 连接。Flash 的地址空间是从 0x00000000～0x00200000。

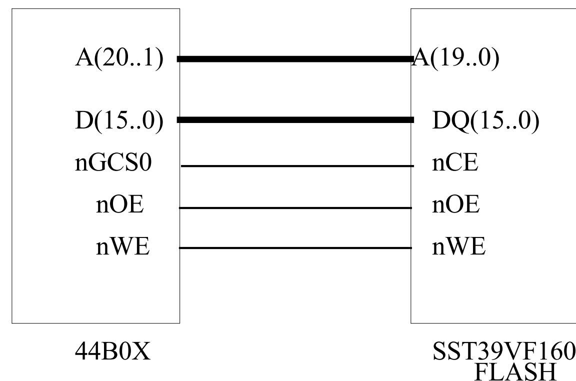
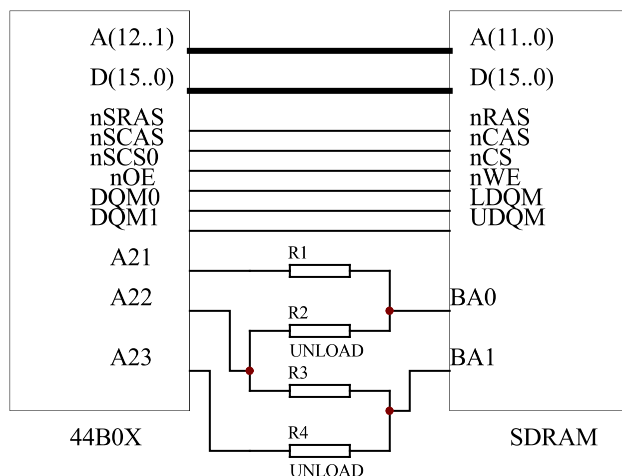


图 3-3 Flash 连接电路

如图 3-4 SDRAM 连接电路所示，SDRAM 分成 4 个 BANK，每个 BANK 的空量为 1M×16bit。BANK 的地址由 BA1、BA0 决定，00 对应 BANK0，01 对应 BANK1，10 对应 BANK2，11 对应 BANK3。在每个 BANK 中，分别用行地址脉冲选通 RAS 和列地址脉冲选通 CAS 进行寻址。本开发板还设置跳线，可以为用户升级内存容量至 4×2M×16bit。对于 8M 的 SDRAM，0 欧姆电阻接在 R1、R3 处，R2、R4 空着，即 BA0、BA1 分别接到 CPU 的 A21、A22 上；行列地址线宽度各为 A1~A11，地址空间为 $4\times2^{10}\times2^{10}$，从 0x0C000000～0x0C3FFFFF。对于 16M 的 SDRAM，0 欧姆电阻接在 R2,R4 处，R1,R3 空着，即 BA0、BA1 分别接到 CPU 的 A22、A23 上；行列地址线宽度各为 A1~A12。地址空间为 $4\times2^{11}\times2^{11}$，从 0x0C000000～0x0C7FFFFF。SDRAM 由 MCU 专用 SDRAM 片选信号 nSCS0 选通，地址空间从 0x0C000000～0x0C8000000。



图中 R1、R2、R3、R4 只同时焊接其中两个：

R1、R3 --- 8MB SDRAM　　R2、R4 --- 16MB SDRAM

图 3-4 SDRAM 连接电路

3.2.3 IIC EEPROM 接口

开发板上提供了一块支持 IIC 总线的 EEPROM（AT24C080），容量为 4Kbit。IIC 是用于内部 IC 控制的简单的双向双线串行总线，在标准模式下数据的传输速度可以达到 100kbit/s，在快速模式下可以达到 400kbit/s。

3.2.4 电源、复位、时钟电路和 JTAG 接口

该开发板是采用 DC 5V 稳压电源进行供电，电源输入后经过板上两个稳压芯片分别产生 3.3V 和 2.5V 的电压，给 MCU 的 I/O 和 ARM 内核供电；

开发板上接了一个系统复位按钮 RESET，按下该按钮即可使系统复位；

实时时钟是通过 MCU 外接 32.768KHz 的晶振来实现的；

JTAG 接口电路如图 3-5 所示，这是 20 脚的标准 JTAG 接口连接电路。

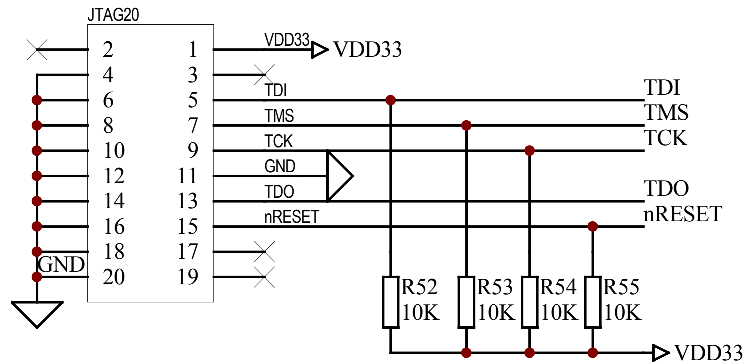


图 3-5 JTAG 接口电路

3.2.5 开关与状态指示灯

JP9 为整个开发板的电源开关，当开关拨到“USBPOWER”时，开发板由 USB 供电；当开关拨到“EXTPOWER”时，由外接电源供电。

D3 为电源指示灯，可以指示目前板子是否上电。

另外以太网口也有 4 个状态指示灯，分别为：

- D5 连接状态指示；
- D6 接收数据指示；
- D13 发送数据指示；
- D14 自检测通过指示。

3.3 Embest EV44B0 通信接口电路

3.3.1 串行接口

如图 3-5 串口电路所示，开发板上提供两个串口 DB9。其中 UART1 为主串口，可与 PC 或 MODOM 进行串行通迅。由于 44B0X 未提供 DCD(载波检测)、DTR（数据终端准备好）、DSR(数据准备好)、RIC（振铃指示）等专用 I/O 口，故用 MCU 的通用 I/O 口替代。UART0 只采用二根接线 RXD 和 TXD，因此只能进行简单的数据传输及接收功能。全接口的 UART1 采用 MAX3243E 作为电平隔离器，简单接口的 UART0 则采用 MAX3221E 作为电平转换器。

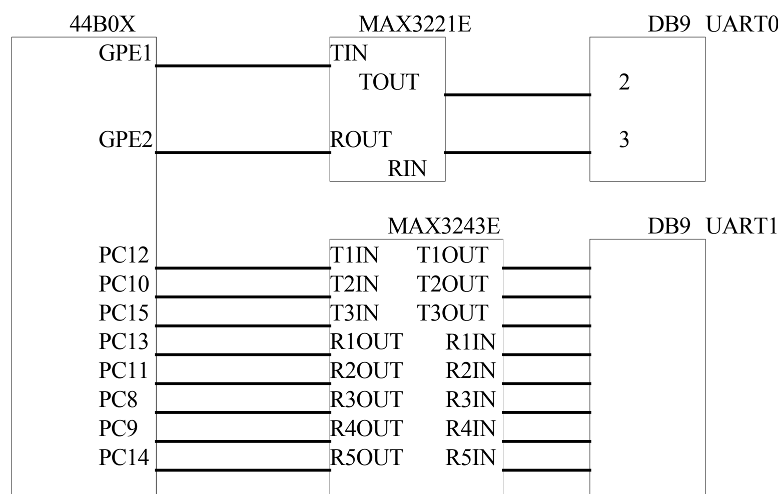


图 3-6 串口电路

3.3.2 USB 电路模块

USB 采用的接口器件是 USBN9603，该 USB 控制器由 NS 公司生产，支持 USB1.0、USB1.1 通信协议，带并行总线。它有 3 种工作模式，即 Non-multiplexed parallel interface mode、Multiplexed parallel interface mode、MICROWIRE interface mode，模式选择由管脚 MODE1、MODE0 决定。设计时把 MODE1，MODE0 接地，因此接口模式定义为无复用并口模式，在该模式中由于 DMA 没有使用，因此把 DACK 接高电平。MCU 通过译码器生成的片选信号 CS1 对 USB 控制器进行选通，USBN9603 通过 EXINT0 对 MCU 发出中断请求。

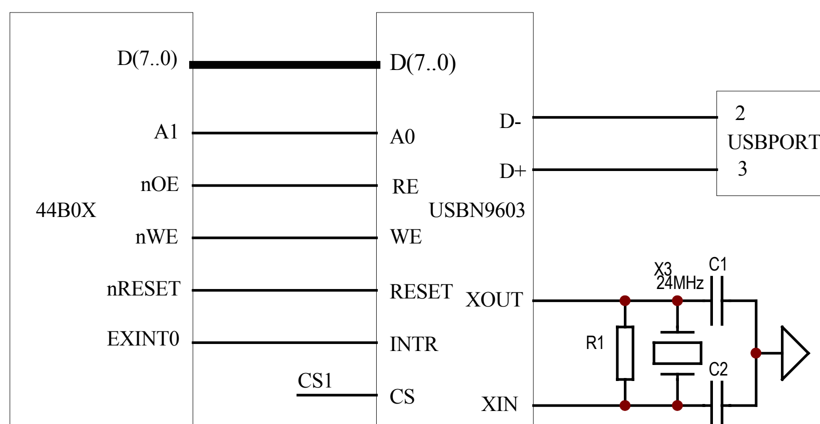


图 3-7 USB 电路模块

3.3.3 Ethernet 电路模块

如图 3-7 以太网电路模块所示，Embest EV44B0 开发板采用 REALTEK 公司生产的、性价比很高的、带有即插即用功能的全双工以太网控制器 RTL8019AS 对 44BOX 进行以太网口扩展。该网络控制器主要特性包括：

- 符合 Ethernet II 与 IEEE802.3 标准；
- 全双工收发可同时达到 10Mbps 的速率；
- 内置 16KB 的 SRAM，用于收发缓冲，降低对主处理器的速度要求；
- 支持 8/16 位数据总线、8 个中断申请线以及 16 个 I/O 基地址选择；
- 支持 UTP，AVI 和 BNC 自动检测，还支持对 10BaseT 拓扑结构的自动极性修正；
- 允许 4 个论断 LED 引脚可编程输出；
- 100 脚的 PQFP 封装，缩小了 PCB 板的尺寸。

RTL8019AS 有三种工作方式，在嵌入式应用中，如果不使用 93C46，可以降低成本，同时又减少连线，应此通常采用跳线工作方式。网卡的 I/O 基址由 IOS3、IOS2、IOS1 和 IOS0 决定。RTL8019AS 中集成了两块 RAM，一块 16KB，地址为 0x4000~0x7FFF；一块 32 字节，地址为 0x0000~0x001F。RAM 按页存储，每 256 字节为一页。一般将第 0 页称为 PROM，用于存放网卡地址，它是网卡复位时从 93C46 里读出来的。但由于本板没有使用 93C46，因此也没有使用 PROM。这时要编程自己指定一个网卡地址，写入到 MAR0~MAR5 内。16KB 的 RAM 则用作收发数据的缓冲区，一般将 0x4000~0x46FF 作为发送缓冲区，0x4700~0x7FFF 作为接收缓冲区。

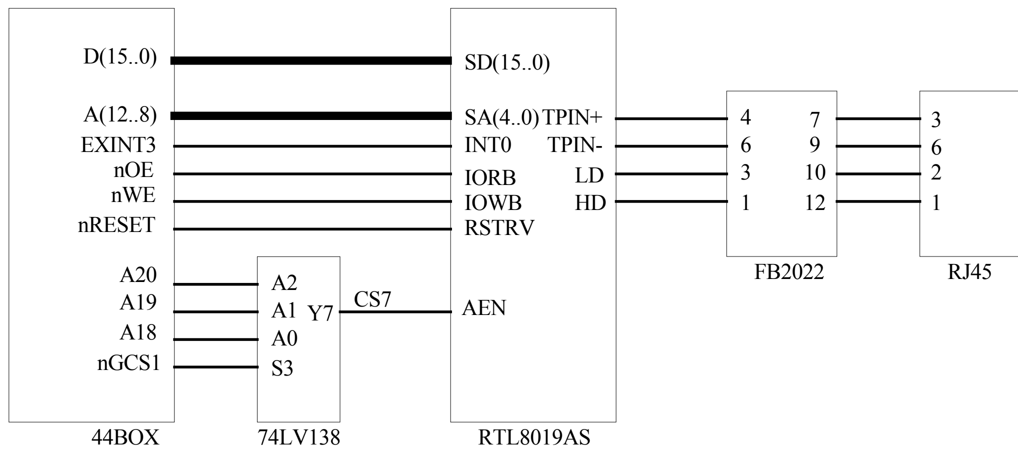

D(15..0)
A(12..8)
EXINT3
nOE
nWE
nRESET
A20
A19
A18
nGCS1
44BOX
A2
A1
A0
S3
Y7
74LV138
CS7
SD(15..0)
SA(4..0)
INT0
IORB
IOWB
RSTRV
AEN
TPIN+
TPIN-
LD
HD
RTL8019AS
4
6
3
1
7
9
10
12
FB2022
3
6
2
1
RJ45

图 3-8 以太网电路模块

3.4 外围扩展模块

3.4.1 IIS 接口

IIS 即音频数据接口，它是 SONY、PHILIPS 等电子巨头共同推出的接口标准。如图 3-8 IIS 接口电路所示，本系统把 IIS 接口与 PHILIPS 的 UDA1341TS 音频数字信号编译码器相连接，得到 MICROPHONE 音频输入通道和 SPEADER 音频输出通道。UDA1341TS 可把立体声模拟信号转化为数字信号，同样也能把数字信号转换成模拟信号，并可用 PGA（可编程增益控制），AGC(自动增益控制)对模拟信号进行处理；对于数字信号，该芯片提供了 DSP(数字音频处理)功能。在实际中，UDA1341TS 可广泛应用于 MD、CD、notebook、PC 和数码摄相机等。44B0X 的 IIS 口可与 UDA1341TS 的 BCK、WS、DATAI、SYSCLK 相连。对于 UDA1341TS 的 L3 总线，它是该芯片工作于微控制器输入模式时使用的，它包括 L3DATA、L3MODE、L3CLOCK 共三根接线，它们分别表示为微处理器接口数据线、微处理器接口模式线，微处理器接口时钟线。通过这个接口，微处理器能够对 UDA1341TS 中的数字音频处理参数和系统控制参数进行配置。但是在 44B0X 中没有设该专用接口，可通过通用 I/O 口进行扩展。

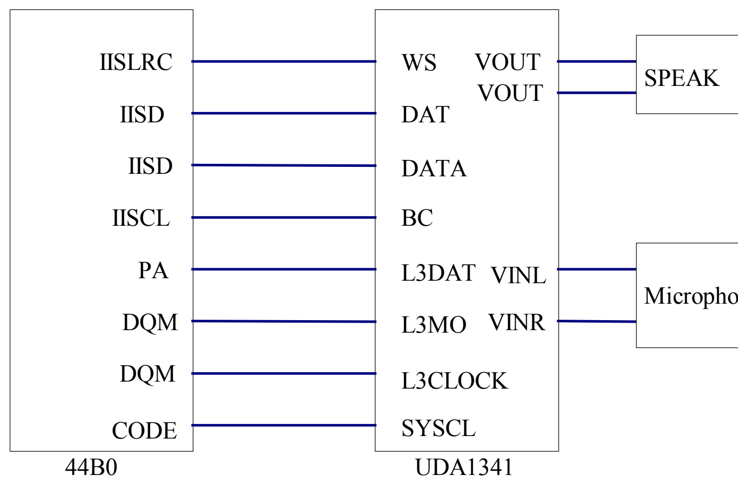


图 3-9 IIS 接口电路

3.4.2 8 段数码管

图 3-9 所示，开发板使用了一个八段数码 LED，该数码管是共阳极的，低电平信号使 LED 点亮。CPU 数据总线 DATA(0~7)经 74LC573 驱动器对数码管进行驱动。其片选信号由 CPU 的 nGCS1 及 3 个地址线 A20、A19、A18 经过译码器生成的 CS6 选通信号选通，而 8-SEG 的内容则由 CPU 低八位数据线决定。

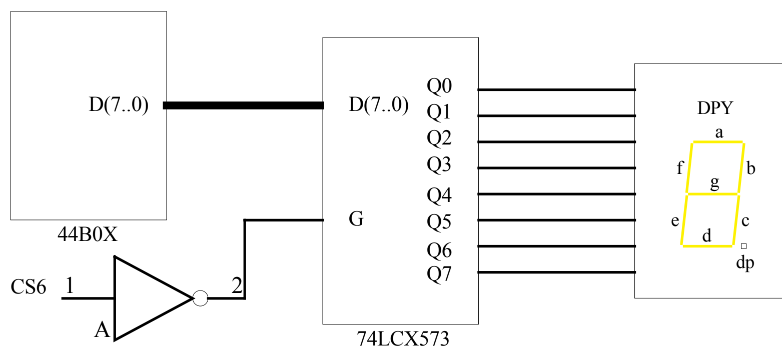


图 3-10 八段数码管

3.4.3 固态硬盘

如图 3-10 所示，Embest EV44B0 开发板提供了一块 16MB 的固态硬盘，芯片型号为 K9F2808。其片选信号为 NGCS1 经 74LV138 译码后得到的 CS2，并用通用 I/O 口 PF6、PF5、NXDACK0、NXDREQ0 分别连接 K9F2808 的 ALE、CLE 、R/B、CE 端口；用户可将固态硬盘与 USB 接口一起当作一个优盘使用，也可将用户自己的程序及数据存储到固态硬盘中。其具体应用有：

- 将采集到的数据存储到固态硬盘中，并可以将这些数据通过 USB 上传到 PC 机上进行备份，分析；
- 可将某系统的参数设置存储于固态硬盘中，并可在系统运行时实时修改，掉电保护等；
- 当系统代码量十分巨大，无法在 2MB 的 FLASH 闪存中运行时，可把代码存放在固态硬盘中，在系统上电时，通过运行在 FLASH 闪存中的启动代码调入大容量的 SDRAM 中运行，此功能对运行大内核操作系统的应用程序十分有用。

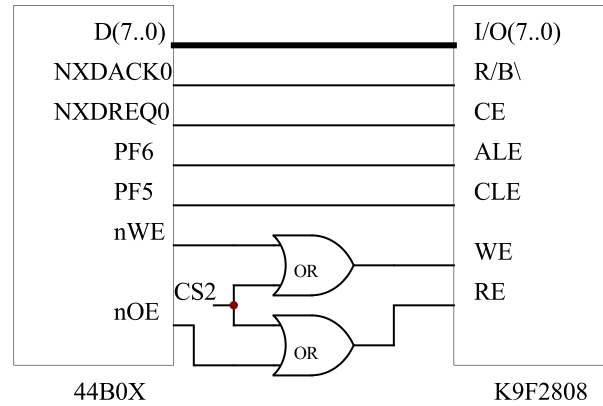


图 3-11 固态硬盘电路模块

3.4.4 IDE 接口

该接口是一个通用的 8bit\16bit 总线扩展口，可挂硬盘或 CF 卡（compact Flash 卡），以及用户自已扩展的外围器件。接口连接到硬盘或 CF 卡时，LED_D4 即硬盘工作指示灯变亮。该接口占用了 CS3、CS4、CS5 三个片选信号及 EXINT4、EXINT5 两个外部中断。

3.4.5 LCD 及 TSP 电路

由于 44B0X 芯片已经提供了 LCD 控制器、驱动器及输入输出口，因此只要把开发板的 LCD 接口相关管脚与 CPU 相应管脚连接即可。44B0X 芯片内集成的 LCD 控制和驱动器可支持单色，4 级灰度，16 级灰度 LCD 及单色、256 色 STN LCD 或 DSTN LCD，典型的实际屏幕尺寸：640×480，320×240，160×160（Pixels），具体 LCD 类型可通过特殊功能寄存器进行配置。LCD 占用的片选信号是 CS8。

至于 TSP，由于 44B0X 芯片未提供该功能，因此可用通用 I/O 口进行配置。TSP 包括两个面电阻，即 X 轴面电阻，Y 轴面电阻，TSP 于是就有 4 个终端口，其接线如图 3-11。在系统处于休眠状态时，Q4、Q2、Q3 处于截止，Q1 为导通；当触摸屏受到笔触时，X 轴面电阻与 Y 轴面电阻在笔触处导通，由于电阻值很小（几百欧左右），使在 EXINT2 处分压得低电平，产生中断；MCU 通过对 I/O 口的控制使 Q2,Q4 导通 Q1,Q3 截止，AIN1 读取 X 轴坐标然后关闭 Q2、Q4，使 Q1、Q3 导通，AIN0 读取 Y 轴坐标。系统行到坐标值后，关闭 Q4、Q2、Q3 打开 Q1，回到初始状态，等待下一次笔触。TSP 占用了 44B0X 外部中断 EXINT2，以及 4 个通用 I/O 口 PE4～PE7。

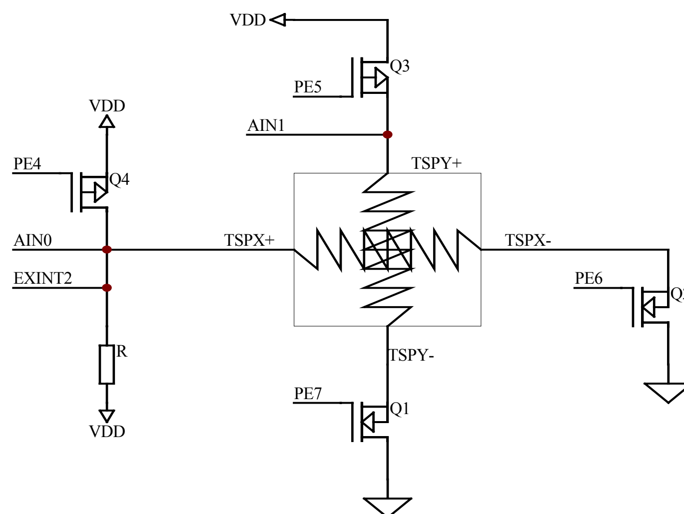


图 3-12 LCD 电路模块

3.4.6　4×4 Keyboard

如图 3-12 键盘接口电路所示，板上扩展了一个 4×4 行列式矩阵键盘接口。该键盘是采用中断扫描的方式进行工作，行线选用了 4 个数据线，列线选用了 4 个地址线。行线接上拉电阻保持高电平，并通过与门 74HC08 将输出信号与 MCU 的中断 EXINT1 连接；列线接下拉电阻保持低电平。当有键盘按下时，该行线被拉为低电平，使得 EXINT1 输入也为低电平，MCU 产生中断。中断产生后通过对键盘的行和列进行扫描的方法可以计算出是哪个键按下，并跳到相应的键盘处理程序中去。芯片 74HC541 是通过片选信号 nGCS3 来选通的，这样可以保证在键盘不使用的情况下 MCU 读不到行线的输入信息。

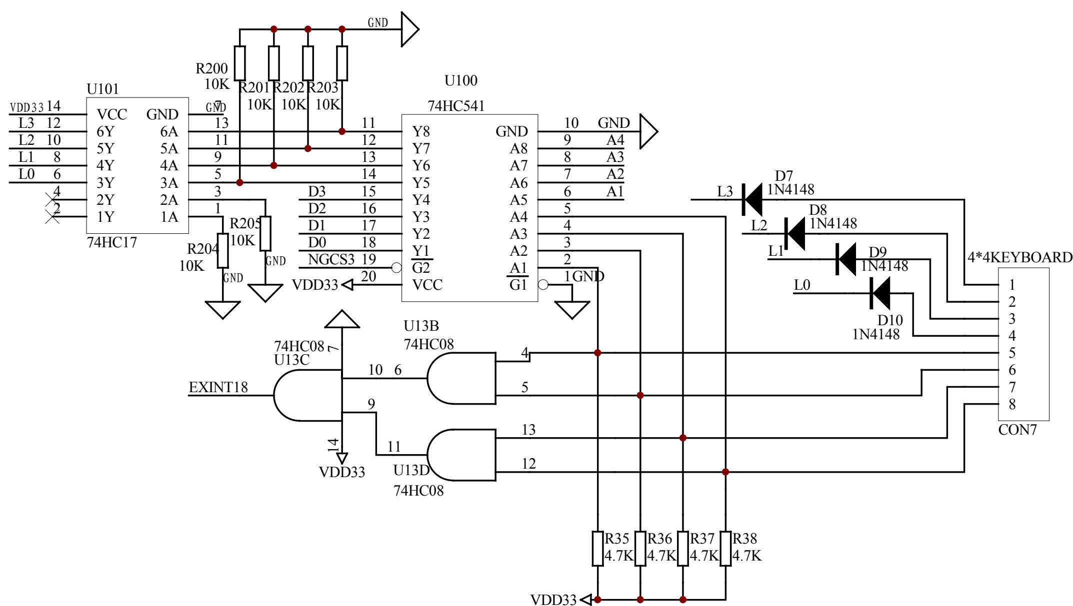


图 3-13 键盘接口电路

3.5 I/O 口分配及结构、地址

3.5.1 片选信号设置

Embest EV44B0 开发板的片选信号设置如表 3-1 所列：

表 **3-1** 片选信号设置

片选信号					选择的接口或器件
NGCS0					FLASH
NGCS6/NSCS0					SDRAM
NGCS1	A20	A19	A18		
	0	0	0	CS1	USB
	0	0	1	CS2	固态硬盘
	0	1	0	CS3	IDE
	0	1	1	CS4	
	1	0	0	CS5	
	1	0	1	CS6	8-SEG
	1	1	0	CS7	ETHERNET
	1	1	1	CS8	LCD

3.5.2 外围地址空间分配

板上外围地址空间为配如表 3-2 所列：

表 **3-2** 外围地址空间分配

外围器件	片选信号	片选控制寄存器	地址空间
FLASH	NGCS0	BANKCON0	0X0000_0000~0X01BF_FFFF
SDRAM	NGCS6	BANKCON6	0X0C00_0000~0X0DF_FFFF
USB	CS1	BANKCON1	0X0200_0000~0X0203_FFFF
固态硬盘	CS2	BANKCON1	0X0204_0000~0X0207_FFFF
IDE(IOR/W)	CS3	BANKCON1	0X0208_0000~0X020B_FFFF
IDE(KEY)	CS4	BANKCON1	0X020C_0000~0X020F_FFFF
IDE(PDIAG)	CS5	BANKCON1	0X0210_0000~0X0213_FFFF
8-SEG	CS6	BANKCON1	0X0214_0000~0X0217_FFFF
ETHERNET	CS7	BANKCON1	0X0218_0000~0X021B_FFFF
LCD	CS8	BANKCON1	0X021C_0000~0X021F_FFFF
NO USE	NGCS2	BANKCON2	0X0400_0000~0X05FF_FFFF
KEYBOARD	NGCS3	BANKCON3	0X0600_0000~0X07FF_FFFF
NO USE	NGCS4	BANKCON4	0X0800_0000~0X09FF_FFFF
NO USE	NGCS5	BANKCON5	0X0A00_0000~0X0BFF_FFFF
NO USE	NGCS7	BANKCON7	0X0E00_0000~0X1FFF_FFFF

3.5.3 I/O 口分配

表 3-3 Port A

Port A	Pin function	Port A	Pin function	Port A	Pin function
PA0	ADDR0	PA4	ADDR19	PA8	ADDR23
PA1	ADDR16	PA5	ADDR20	PA9	OUTPUT(IIS)
PA2	ADDR17	PA6	ADDR21		
PA3	ADDR18	PA7	ADDR22		

PCONA 寄存器地址：0X01D20000

PDATA 寄存器地址：0X01D20004

PCONA 复位默认值：0X1FF

表 3-4 Port B

Port B	Pin function	Port B	Pin function	Port B	Pin function
PB0	SCKE	PB4	OUTPUT(IIS)	PB8	NGCS3
PB1	SCLE	PB5	OUTPUT(IIS)	PB9	OUTPUT(LED1)
PB2	nSCAS	PB6	nGCS1	PB10	OUTPUT(LED2)
PB3	nSRAS	PB7	NGCS2		

PCONB 寄存器地址：0X01D20008

PDATB 寄存器地址：0X01D2000C

PCONB 复位默认值：0X7FF

表 3-5 Port C

Port C	Pin function	Port C	Pin function	Port C	Pin function
PC0	IISLRCK	PC6	VD5	PC12	TXD1
PC1	IISDO	PC7	VD4	PC13	RXD1
PC2	IISDI	PC8	INPUT（串口）	PC14	INPUT（串口）

PC3	IISCLK	PC9	INPUT（串口）	PC15	INPUT（串口）
PC4	VD7	PC10	RTS1		
PC5	VD6	PC11	CTS1		

PCONC 寄存器地址：0X01D20010

PDATC 寄存器地址：0X01D20014

PUPC 寄存器地址：0X01D20018

PCONC 复位默认值：0X0FF0FFFF

表 3-6 Port D

Port D	**Pin function**	**Port D**	**Pin function**	**Port D**	**Pin function**
PD0	VD0	PD3	VD3	PD6	VM
PD1	VD1	PD4	VCLK	PD7	VFRAME
PD2	VD2	PD5	VLINE		

PCOND 寄存器地址：0X01D2001C

PDATD 寄存器地址：0X01D20020

PUPD 寄存器地址：0X01D20024

PCOND 复位默认值：0XAAAA

表 3-7 Port E

Port E	**Pin function**	**Port E**	**Pin function**	**Port E**	**Pin function**
PE0	OUTPUT(LCD)	PE3	RESERVE	PE6	OUTPUT(TSP)
PE1	TXD0	PE4	OUTPUT(TSP)	PE7	OUTPUT(TSP)
PE2	RXD0	PE5	OUTPUT(TSP)	PE8	CODECLK

PCONE 寄存器地址：0X01D20028

PDATE 寄存器地址：0X01D2002C

PUPE 寄存器地址：0X01D20030

PCONE 复位默认值：0X25529

表 **3-8 Port F**

Port F	Pin function	Port F	Pin function	Port F	Pin function
PF0	IICSCL	PF3	IN（固态硬盘）	PF6	out(固态硬盘)
PF1	IICSDA	PF4	out（固态硬盘）	PF7	IN(bootloader)
PF2	RESERVED	PF5	out(固态硬盘)	PF8	IN(bootloader)

PCONF 寄存器地址：0X01D20034

PDATF 寄存器地址：0X01D20038

PUPF　寄存器地址：0X01D2003C

PCONF 复位默认值：0X00252A

表 **3-9 Port G**

Port G	Pin function	Port G	Pin function	Port G	Pin function
PG0	EXINT0	PG3	EXINT3	PG6	EXINT6
PG1	EXINT1	PG4	EXINT4	PG7	EXINT7
PG2	EXINT2	PG5	EXINT5		

PCONG 寄存器地址：0X01D20040

PDATG 寄存器地址：0X01D20044

PUPG　寄存器地址：0X01D20048

PCONG 复位默认值：0XFFFF

3.6　总线扩展

Embest EV44B0 开发板上预留了所有引脚的扩展接口，用户可以非常方便地根据需求扩展存储器和各种外部设备，能够满足大部分产品的应用。用户在扩展时需要自己制作扩展板，只要扩展板上的接口定义和开发板上扩展接口相对应即可。

第四章 Embest S3C44B0 开发板软件系统

4.1 软件开发调试与程序固化

4.1.1 软件开发

为配合用户快速使用开发板进行产品的软件开发，公司在开发板出厂时提供了丰富的运行例程，用户可把任何一个样例程序作为软件开发模板。并可藉此结合实际的硬件环境编写软件程序，使开发周期大大的缩短。

用户进行软件开发时应注意：

1) 如果软件在 RAM 中调试，则存储区的配置将由集成环境通过命令脚本文件完成，因此在程序中不需要使用到存储区配置代码；
2) 在 RAM 中调试时，数据段内容不需要拷贝，这在程序中通过判断只读区域和可读写区域地址是否重叠自动选择；
3) 代码中对异常向量如预取失败并未做任何处理，完善的程序应该处理任一个异常向量，包括保存进入异常向量前的执行状态以备查询，清除异常可能产生的错误后返回程序继续执行。

4.1.2 软件调试

RAM 调试

Embest IDE for ARM 为用户进行软件开发提供了两种调试方法：RAM 调试和 Flash 调试。由于 RAM 区可以很方便地读写，访问速度高，因此软件开发过程中的用户程序调试只要硬件条件许可，都应该在 RAM 区完成。灵活地运用 RAM 调试和 Flash 调试方法，可以快速发现和处理项目开发过程中许多复杂的程序设计问题，加快项目开发的进度。

在 Embest IDE for ARM 开发环境中进行软件调试前需要完成以下几步：编译链接工程，连接仿真器、电路板，程序下载。

- 编译链接工程

用户选择生成（Build）菜单，编译相应的文件或工程，在输出（Output）窗口的生成（Build）子窗口中输出相应的编译、链接信息。按照链接配置，程序编译通过以后在工程目录\debug\目录下生成 l.elf 文件，该文件是包含调试信息的执行文件。

- 连接仿真器、开发板

选择调试（Debug）菜单的远程连接（Remote Connect）子菜单，集成环境中的调试器通过仿真器和目标系统相连接。

- 程序下载

目标系统连接后，如果在调试配置选项中设置了自动下载选项，调试器将自动下载软件；否则选择菜单调试（Debug）的下载（Download）子菜单下载程序。此时，调试器将.elf中的调试信息去除后下载二进制指令文件到目标板存储区指定的位置，同时在状态条上显示下载进度。下载地址是经命令脚本映射的 RAM 存储区起始地址。下载成功后，状态条以蓝色状态条显示“Download Completed”信息，否则以红色状态条显示“Download Failed”信息。

Flash 调试

当电路板由于硬件资源限制，比如 RAM 区空间小于程序代码空间而不能在 RAM 区调试，或者需要观察应用程序在实际硬件环境中运行的情况时，可以把在 RAM 调试通过的应用程序 Bin 代码文件烧写到 Flash 芯片中进行调试。烧写到 Flash 芯片的程序运行不能得到正确的结果或观察程序在 Flash 中运行情况，用户都需要进行 Flash 调试工作。

程序在 Flash 中调试与在 RAM 中调试工程配置不同：

- 调试选项中不需要执行脚本文件，该工作在启动文件中完成，需要将连接后行为（Action after connected）选项改为无（None）；

调试过程也有所不同：

- 连接仿真器后，无需再执行下载（Download）程序操作；
- 如果要从启动程序的入口开始调试程序，先必须执行复位（reset）命令，此时程序将停在零地址处；
- 程序在 Flash 中调试时最多可以设置两个硬件断点。

4.1.3 程序固化

在 RAM 中调试通过的程序与最终固化到电路板的 Flash 中的程序有所区别，用户需要:

- 在汇编器的预定义选项中设置编译定义符号，或者直接在初始化文件中增加 定义编译项，由启动文件自己完成存储区的重映射而不是由命令脚本完成。
- 在链接器的链接文件中选择 flash.ld，该链接文件和启动文件配合完成最初下载到 Flash 中的数据段的搬运工作。

完成以上改动后，重新编译程序。然后使用 Elf to Bin 工具将.elf 文件转换成二进制指令格式文件.bin。再利用 Embest Flash Programmer 工具将.bin 下载到电路板的 Flash 中。

4.2 启动程序介绍

启动程序设计应注意开发板所使用的处理器 S3C44B0 没有存储区重映射功能，所有存储区地址固定，另外 S3C44B0 提供向量中断功能，减少了中断延迟。因此在启动程序设计时，向量中断功能的出现导致扩展了向量表，同时为方便程序设计和在 RAM 中调试，中断入口通过地址定义方式转移到 RAM 区的最高端。

下面的代码是 S3C44B0 的启动程序源代码及其解释，部分相似的中断入口定义和函数宏定义被省略，省略部分以……代替并加上了注释，用户如果要使用下面的源代码作为启动程序时，可以参考相关数据手册进行修改或自行添加省略部分程序。

```
# ******************************************************************
# 文件名: 44bINIT.S                                                *
# 说明:    S3c44b0x 启动文件                                        *
# ******************************************************************

#===============================================================
# 寄存器定义及其位定义
#===============================================================
.equ    INTMSK,         0x01e0000c
.equ    WTCON,          0x01d30000

.equ    CLKCON,         0x01d80004
.equ    LOCKTIME,       0x01d8000c

.equ    FIQMODE,        0x11
.equ    IRQMODE,        0x12
.equ    SVCMODE,        0x13
.equ    ABORTMODE,      0x17
.equ    UNDEFMODE,      0x1b
.equ    MODEMASK,       0x1f
.equ    NOINT,          0xc0
.equ    CPSR_IRQ_EN,    0x80
#===============================================================
# 中断处理宏
#===============================================================
    .macro HANDLER HandleLabel
```

```
    sub     sp, sp, #4              @ 栈空间递减保存跳转地址
    stmfd   sp!, {r0}               @ 保存工作寄存器 r0 到栈
    ldr     r0, =\HandleLabel       @ 载入中断入口地址所在位置到 r0
    ldr     r0, [r0]                @ 载入中断入口地址到 r0
    str     r0, [sp,#4]             @ 保存中断入口地址到栈
    ldmfd   sp!, {r0,pc}            @ 恢复工作寄存器并跳转到中断函数
    .endm

#==================================================
# 设置 ARM7 中断和异常向量
#==================================================
ENTRY:
    b   ResetHandler                @ S3C4510 复位后从此处执行
    b   HandlerUndef                @ 未定义异常向量
    b   HandlerSWI                  @ 软中断向量
    b   HandlerPabort               @ 取指异常向量
    b   HandlerDabort               @ 取数据异常向量
    b   .                           @ 保留
    b   HandlerIRQ                  @ 中断向量
    b   HandlerFIQ                  @ 快速中断向量

#==================================================
# 设置 44B0 中断向量表
#==================================================
VECTOR_BRANCH:
    ldr pc,=HandlerEINT0            @ mGA H/W interrupt vector table
    ldr pc,=HandlerEINT1            @
    ......                          @ 省略
    ldr pc,=HandlerADC              @ mGKB
    ......                          @ 省略
    b .

#==================================================
# 中断向量处理宏
#==================================================
HandlerFIQ:        HANDLER HandleFIQ
HandlerIRQ:        HANDLER HandleIRQ
```

```
HandlerUndef:       HANDLER HandleUndef
HandlerSWI:         HANDLER HandleSWI
HandlerDabort:      HANDLER HandleDabort
HandlerPabort:      HANDLER HandlePabort

HandlerADC:         HANDLER HandleADC
    ......                              @ 省略
HandlerEINT1:       HANDLER HandleEINT1
HandlerEINT0:       HANDLER HandleEINT0

#================================================
# 中断向量处理宏
#================================================
ResetHandler:
    ldr     r0,=WTCON                   @ 看门狗禁止
    ldr     r1,=0x0
    str     r1,[r0]

    ldr     r0,=INTMSK
    ldr     r1,=0x07ffffff              @ 所有中断禁止
    str     r1,[r0]

#================================================
# 设置时钟控制控制器
#================================================
    ldr     r0, =LOCKTIME
    ldr     r1, =0xfff
    str     r1, [r0]

    ldr     r0, =CLKCON
    ldr     r1, =0x7ff8                 @ 所有模块的时钟开启
    str     r1, [r0]

#================================================
# 设置存储区控制器
#================================================
    ldr     r0, =SMRDATA
```

```
    ldmia   r0, {r1-r13}
    ldr     r0, =0x01c80000
    stmia   r0, {r1-r13}

#==========================================================
# 初始化栈空间
#==========================================================
    ldr     sp, =SVCStack           @ 切换到超级用户栈空间
    bl      InitStacks

#==========================================================
# 引入外部符号，符号定义在链接脚本文件中
#==========================================================
    .extern  Image_RO_Limit         @ 只读区域大小
    .extern  Image_RW_Base          @ 可读写存储区域起始地址
    .extern  Image_ZI_Base          @ 清零区域起始地址
    .extern  Image_ZI_Limit         @ 清零区域大小

#==========================================================
# 初始化 C 代码需要使用的存储区
#==========================================================
    LDR     r0, =Image_RO_Limit     @ 获取只读区域大小
    LDR     r1, =Image_RW_Base      @ 获取可读写区域起始地址
    LDR     r3, =Image_ZI_Base      @ 获取清零区域起始地址
    CMP     r0, r1                  @ 比较只读区域和可读写区域是否重叠
    BEQ     LOOP1
LOOP0:
    CMP     r1, r3                  @ 拷贝程序中.data 数据段内容到读写区域
    LDRCC   r2, [r0], #4
    STRCC   r2, [r1], #4
    BCC     LOOP0
LOOP1:
    LDR     r1, =Image_ZI_Limit     @ 从清零区域顶部开始
    MOV     r2, #0
LOOP2:
    CMP     r3, r1                  @ 清零
    STRCC   r2, [r3], #4
```

```
    BCC     LOOP2
#=============================
# 全能 IRQ 中断（I 位）
#=============================
    MRS     r0, CPSR
    BIC r0, r0, #CPSR_IRQ_EN /* IRQ enable */
    MSR     CPSR_cxsf, r0
#=============================
# 进入 C 语言程序入口
#=============================
   .extern __main
   BL       __main

#==================================================
# 初始化栈空间的函数
#==================================================
InitStacks:
    mrs     r0, cpsr
    bic     r0, r0, #MODEMASK
    orr     r1, r0, #UNDEFMODE | NOINT
    msr     cpsr_cxsf, r1
    ldr     sp, =UndefStack           @ 设置未定义异常栈空间

    orr     r1, r0, #ABORTMODE|NOINT
    msr     cpsr_cxsf, r1
    ldr     sp, =AbortStack           @ 设置异常栈空间

    orr     r1, r0, #IRQMODE|NOINT
    msr     cpsr_cxsf, r1
    ldr     sp, =IRQStack             @ 设置中断栈空间

    orr     r1, r0, #FIQMODE|NOINT
    msr     cpsr_cxsf, r1
    ldr     sp, =FIQStack             @ 设置快速中断栈空间

    bic     r0, r0, #MODEMASK|NOINT
    orr     r1, r0, #SVCMODE
```

```
    msr     cpsr_cxsf, r1
    ldr     sp, =SVCStack              @ 设置超级用户栈空间

    mov         pc,lr                  @ 函数返回

#==========================================================
# 存储区相关寄存器设置值
#==========================================================
SMRDATA:
    .long 0x11110101                   @ 存储区访问宽度控制寄存器
    .long 0x00000600                   @ BANK0 控制寄存器
    .long 0x00007FFC                   @ BANK1 控制寄存器
    .long 0x00007FFC                   @ BANK2 控制寄存器
    .long 0x00007FFC                   @ BANK3 控制寄存器
    .long 0x00007FFC                   @ BANK4 控制寄存器
    .long 0x00007FFC                   @ BANK5 控制寄存器
    .long 0x00018000                   @ BANK6 控制寄存器
    .long 0x00018000                   @ BANK7 控制寄存器
    .long 0x00860459                   @ SDRAM 刷新控制寄存器
    .long 0x10                         @ SDRAM 存储区大小
    .long 0x20                         @ BANK6 SDRAM 模式寄存器
    .long 0x20                         @ BANK7 SDRAM 模式寄存器

.equ    STARTADDRESS,    0xc7fff00
#==========================================================
# 栈空间定义
#==========================================================
.equ    UserStack,      STARTADDRESS-0x500          @ c1(c7)ffa00
.equ    SVCStack,       STARTADDRESS-0x500+256      @ c1(c7)ffb00
.equ    UndefStack,     STARTADDRESS-0x500+256*2    @ c1(c7)ffc00
.equ    AbortStack,     STARTADDRESS-0x500+256*3    @ c1(c7)ffd00
.equ    IRQStack,       STARTADDRESS-0x500+256*4    @ c1(c7)ffe00
.equ    FIQStack,       STARTADDRESS-0x500+256*5    @ c1(c7)fff00

#==========================================================
# ARM 中断向量入口定义
#==========================================================
```

```
.equ    HandleReset,        STARTADDRESS
.equ    HandleUndef,        STARTADDRESS+4
.equ    HandleSWI,      STARTADDRESS+4*2
.equ    HandlePabort,       STARTADDRESS+4*3
.equ    HandleDabort,       STARTADDRESS+4*4
.equ    HandleReserved,     STARTADDRESS+4*5
.equ    HandleIRQ,      STARTADDRESS+4*6
.equ    HandleFIQ,      STARTADDRESS+4*7

#=================================================
# S3C44B0 中断向量入口定义
#=================================================
.equ    HandleADC,      STARTADDRESS+4*8
......                                          @ 省略
.equ    HandleEINT4567,     STARTADDRESS+4*29
.equ    HandleEINT3,    STARTADDRESS+4*30
.equ    HandleEINT2,    STARTADDRESS+4*31
.equ    HandleEINT1,    STARTADDRESS+4*32
.equ    HandleEINT0,    STARTADDRESS+4*33           @ 0xc1(c7)fff84
```

4.3 µCOS-II

4.3.1 概 述

µCOS-II 是µCOS 的升级版本。µCOS-II 产品在现代商业应用非常广泛，已被相关机构证实具有非常稳定、可靠的性能，并成功应用于生命科学、航天工程等重大科研项目中。并且由于其极小的内核,特别适用于对程序代码存储空间要求极其敏感的嵌入式系统开发。

µCOS-II 是一款源码公开的实时操作系统，真正支持多个任务同时运行，各个任务有独立的栈空间，并提供系统服务、中断管理等功能。作一个实用的实时操作系统，µCOS-II 还具有以下特点：

- 可移植性（Portable）
- 可固化（ROMable）
- 可裁剪（Scalable）
- 抢占式（Preemptive）

教学系统已成功运行的µCOS-II 系统是 Embest IDE for ARM 工具编译调试通过的简单内核和用户程序，使用 Embest IDE for ARM 可以查看内核的各种状态，包括任务（Task）、队列（Queue）、信号量（Semaphore）、邮箱（Mailbox）、事件（Event）、互斥量（Mutex）等；同时提供时间相关函数、栈空间、内存（Memory）申请和释放等操作。

4.3.2 µCOS-II 调试及开发指导

用户可以直接使用 Embest IDE for ARM 附带已配置的工程文件，进行µCOS-II 系统下的软件开发。打开位于开发板光盘\Software\S3CEV40\uCOS-II 文件夹下的工作区文件（.ews）。如图 4-1 所示。

在进行µCOS-II 应用程序开发前，希望用户先参考一些有关µCOS-II 使用的书藉，这样对用户进行µCOS-II 环境下的编程有一定的帮助。按照 Embest IDE for ARM 提供的µCOS-II 例程，用户只需在 main.c 文件进行相应的任务修改，编写相应的应用函数即可。下面以在 Embest IDE for ARM 开发环境下为µCOS-II 增加一个运行任务为例，说明嵌入式实时操作系统下的应用程序开发。

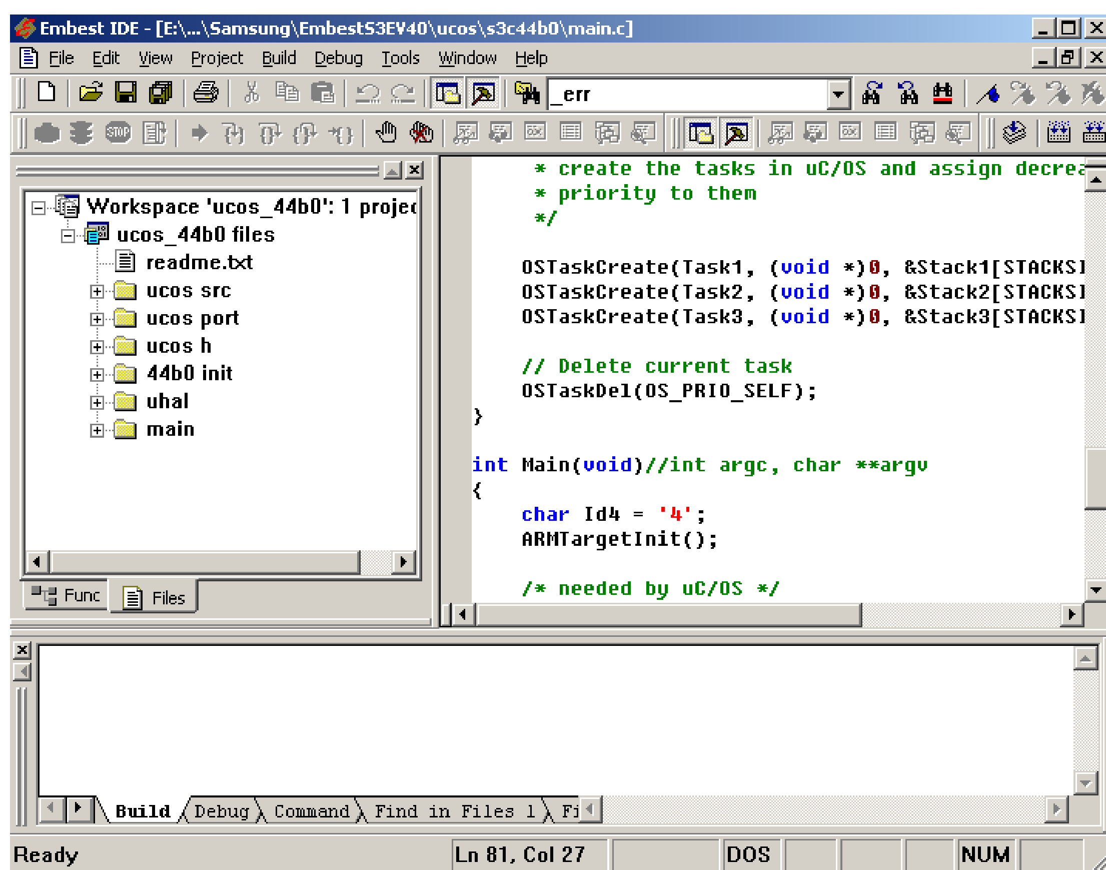


图 4-1 打开μCOS-II for EmbestS3CEV40 工程

打开 main.c 文件，按照增减μCOS-II 运行任务的编程步骤修改程序，即：

分配任务栈（**unsigned int Stack4[STACKSIZE];**）

为应用程序运行时的变量、堆栈提供存放和访问空间

```
unsigned int Stack4[STACKSIZE];
```

建立任务函数体（**void Task4(void *Id)**）

```
void Task4(void *Id)
{
    变量定义及初始化
    功能函数或指令语句
    OSTimeDly(100);//任务挂起时间间隔
}
```

启动任务描述（**OSTaskCreate(Task4, (void *)0, &Stack4[STACKSIZE - 1], 5);**）

```
void TaskStart (void *Id)
{
    ……
    char Id4 = '4';
    ……
    OSTaskCreate(Task4, (void *)0, &Stack4[STACKSIZE - 1], 5);
    ……
}
```

在 **main()**函数内加入要初始化的模块函数

经以上修改后的 main.c 文件内容如下：

```
#include  "../../uCOS-II /includes.h"                    /* uC/OS interface */
//task stack size
#ifdef SEMIHOSTED
    #define   TASK_STACK_SIZE    (64+SEMIHOSTED_STACK_NEEDS)
#else
    #define          TASK_STACK_SIZE    10*1024
#endif
//Task definition
/* allocate memory for tasks' stacks */
#define STACKSIZE 64
/* Global Variable */
unsigned int Stack1[STACKSIZE];
unsigned int Stack2[STACKSIZE];
unsigned int Stack3[STACKSIZE];
```

```
unsigned int Stack4[STACKSIZE];
unsigned int StackMain[STACKSIZE];
void Task1(void *Id)
{
    while(1)
    {
        leds_on();
        OSTimeDly(300);
        leds_off();
        OSTimeDly(130);
    }
}
void Task2(void *Id)
{
    int i;
    while(1)
    {
        for(i=0; i<16; i++)
          {
            Digit_Led_Symbol(i);
            OSTimeDly(150);
          }
    }
}
void Task3(void *Id)
{
```

```
        int i=0;
        while(1)
        {
                i++;
                uHALr_printf("                    **** %d ****\r",i);
            OSTimeDly(90);
        }
}
void Task4(void *Id)
{
        int i=0;
        while(1)
        {
                i++;
                //user function
            OSTimeDly(70);
        }
}
void TaskStart (void *i)
{
         char Id1 = '1';
         char Id2 = '2';
         char Id3 = '3';
         char Id4 = '4';

         uHALr_InitTimers();                    // enable timer counter interrupt
```

```
        /*
        * create the tasks in uC/OS and assign decreasing
        * priority to them
        */
       OSTaskCreate(Task1, (void *)&Id1, &Stack1[STACKSIZE - 1], 2);
       OSTaskCreate(Task2, (void *)&Id2, &Stack2[STACKSIZE - 1], 3);
       OSTaskCreate(Task3, (void *)&Id3, &Stack3[STACKSIZE - 1], 4);
       OSTaskCreate(Task4, (void *)&Id4, &Stack4[STACKSIZE - 1], 5);
       // Delete current task
       OSTaskDel(OS_PRIO_SELF);
}
int Main(void)//int argc, char **argv
{
       char Id0 = '0';
       ARMTargetInit();
       /* needed by uC/OS */
       OSInit();
       OSTimeSet(0);
       /* create the start task */
       OSTaskCreate(TaskStart,(void *)&Id0, &StackMain[STACKSIZE - 1], 0);
       ARMTargetStart();
       /* start the operating system */
       OSStart();
       return(0);
//* End
}
```

在工作区窗口加入应用程序文件

在工作区窗口中建立应用程序文件夹后加入相应的文件。如图 4-2 所示。

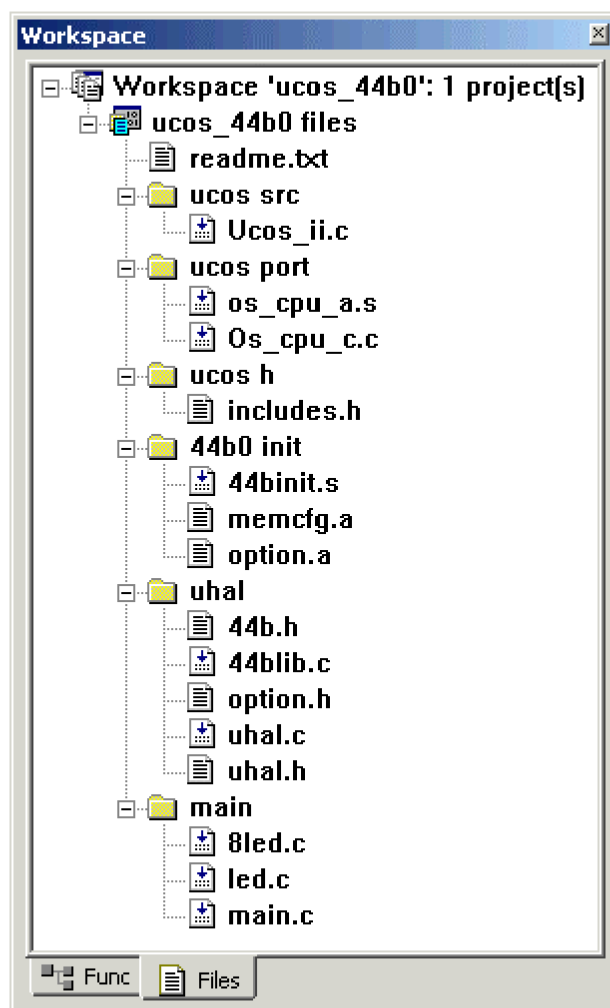


图 4-2 uCOS-II 工程的工作区窗口

编译链接成功后把调试文件下载到目标板的 RAM 中调试

工程下载后，程序运行到 main()的 Embest IDE for ARM 的界面如图 4-3 所示。

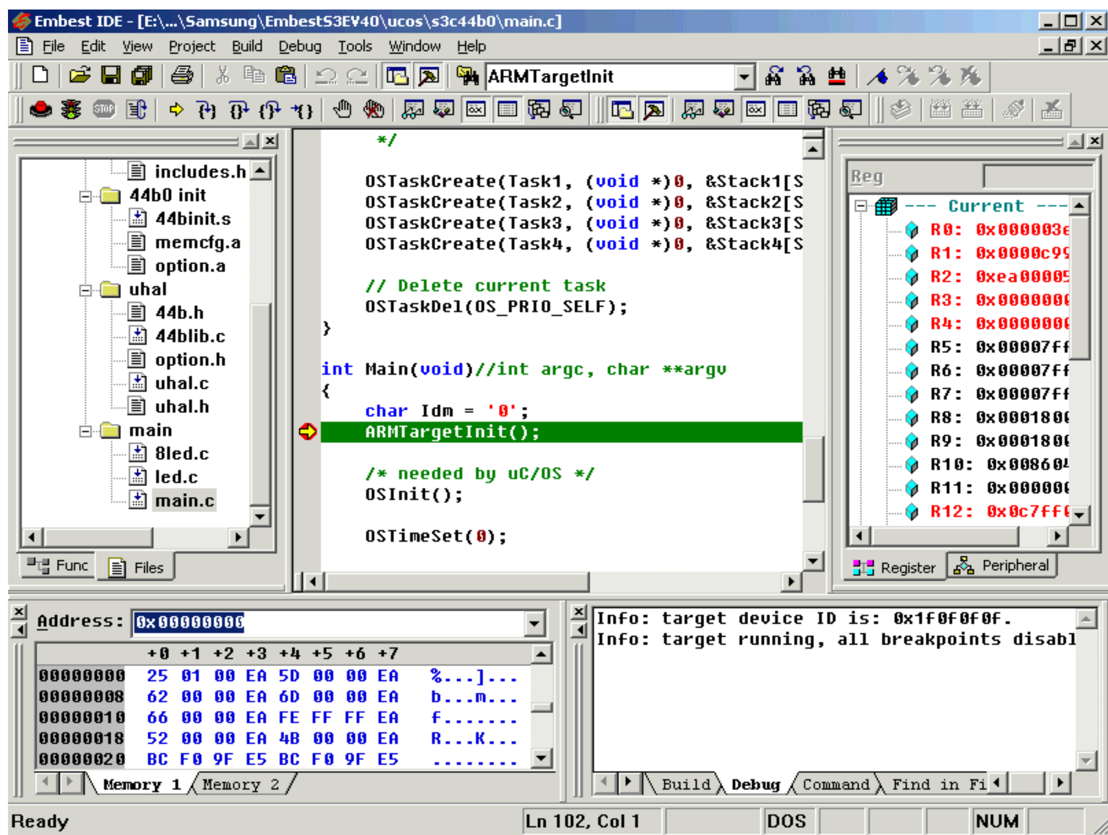


图 4-3 uCOS-II 增加用户任务后调试运行界面

调试结果正确后利用 Elf2Bin 把调试文件转换成.bin 代码文件，使用 Embest Flash Programmer 工具把.bin 文件烧写到目标板上观察运行结果。

整个开µCOS-II 操作系统的应用程序开发过程到此就完成了。

4.4 Example Codes

例一、 定时器使用示例

工作区文件位于：开发板光盘\Software\S3CEV40\timer_test

例程功能：使用户掌握定时器的工作方式、控制寄存器使用。

例二、 I/O 接口使用示例

工作区文件位于：开发板光盘\Software\S3CEV40\led_test

例程功能：通过对处理器的 I/O 口的配置及使用，掌握 I/O 编程技巧。

例三、 RS232 串口收发通信示例

工作区文件位于：开发板光盘\Software\S3CEV40\UART_test

例程功能：通过交互输入完成处理器与 PC 机的通信控制，掌握串口编程控制方法

例四、 七段数码管的使用示例

工作区文件位于：开发板光盘\Software\S3CEV40\8Led_test

例程功能：掌握处理器地址数据线的定义及使用

例五、 4 x 4 键盘使用示例

工作区文件位于：开发板光盘\Software\S3CEV40\keyboard_test

例程功能：掌握处理器地址数据线的定义及使用

例六、 IIC 接口 E²PROM 读写示例

工作区文件位于：开发板光盘\Software\S3CEV40\iic_test

例程功能：实现串行信号读写 E2PROM 器件、提供读写控制程序编写方法

例七、 IIS 接口声音控制示例

工作区文件位于：开发板光盘\Software\S3CEV40\iis_test

例程功能：实现音频接口信号编程及输出、掌握嵌入式处理器 IIS 接口编程方法。

例八、 LCD 液晶显示控制示例

工作区文件位于：开发板光盘\Software\S3CEV40\lcd_test

或开发板光盘\Software\S3CEV40\Bmp_Display

例程功能： 掌握 S3C44b0 处理器 LCD 驱动器的使用。提供包括象素点控制、颜色控制、字库以及画点、画线等大量控制程序代码。

例九、 触摸屏使用控制示例

工作区文件位于：开发板光盘\Software\S3CEV40\ TouchScreen_Test

例程功能：处理器的高级编程应用。初步掌握人机接口工作原理和编程方法。

例十、 **TFTP** 控制协议实现示例

工作区文件位于：开发板光盘\Software\S3CEV40\tftp_test

例程功能：例程通过网卡芯片控制与编程，实现以太网口的文件传送，并且把数据文件存放到 Flash 芯片。

例十一、 **DHCP** 控制协议实现示例

工作区文件位于：开发板光盘\Software\S3CEV40\dhcp_test

例程功能：例程通过网卡芯片控制与编程，完成动态 IP 配置。实现了 UDP 传输协议。

例十二、 **µCOS-II** 操作系统使用示例

工作区文件位于：开发板光盘\Software\S3CEV40\uCOS-II

例程功能：提供学习和使用实时操作系统，掌握操作系统下应用程序的编程方法。

第五章　售后服务与技术支持

深圳市英蓓特信息技术有限公司承诺为我们的客户提供相关技术支持。如果您在使用我公司产品的时候，遇到任何问题，可以通过下列途径与我们客户服务部的技术支持工程师联系：

- 英蓓特公司网站

关于英蓓特公司产品的最新最准确的信息（包括公司产品信息以及相关资料），您可以通过以下网址得到：http://www.embedinfo.com。

- 技术论坛

英蓓特公司提供两个主力论坛供我们的广大客户以及业界的工程师相互交流和学习。

ARM 开发论坛：

讨论 ARM 技术、ARM 系列芯片、ARM 开发工具、ARM 嵌入式处理器的开发、ARM 应用的论坛。

Embest IDE 用户论坛：

Embest IDE 用户技术交流、技术支持的论坛。

- 邮件

用户可以通过邮件地址 support@embedinfo.com 直接与我们的客户服务工程师联系。

- 电话

用户可以在工作时间拨打我们的客户服务热线电话 **86-755-25631365**。

- 传真

用户如有相关资料需要传真，您可以使用这个号码 **86-755-25616057**。

Embest S3CEV40 开发板，整体保修期为 3 个月，但电源、电缆等易耗件及人为损坏不在保修范围之内。

警告：

请务必注意静电的防护。超过任何最大承受值，均会对产品产生永久损害。同时，不推荐在临界状态使用产品。

附录 A 跳线与开关设置

表 **A-1 SW1** 和 **SW2** 设置

SW1、SW2（跳线）	功能
全部连接	串口 UART1 可以正常使用
断开	把 UART1 使用的 I/O 口保留下来供用户使用

表 **A-2 SW3** 设置

SW3（跳线）	功能
连接	该跳线保留，暂时未使用
断开	

表 **A-3 SW4** 设置

SW4（跳线）	功能
连接	该跳线保留，暂时未使用
断开	

表 **A-4 SW5** 设置

SW5（跳线）	功能
连接	选择 Little endian 模式
断开	选择 Big endian 模式

表 **A-5 SW6** 设置

SW6（开关）	功能
USBPOWER	开发板通过 USB 来供电
EXTPOWER	开发板通过外接 5V 直流稳压电源供电